The contribution of context information: a case study of object recognition in an intelligent car

Alexander Gepperth, Benjamin Dittes, Michaël Garcia Ortiz

The contribution of context information: a case study of object recognition in an intelligent car

Alexander Gepperth[a,*], Benjamin Dittes[b], Michaël Garcia Ortiz[c]

[a] *ENSTA ParisTech, 32 Blvd Victor, 75739 Paris Cedex 15, France*
[b] *Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany*
[c] *CoR-Lab, Universität Bielefeld, Universitätsstr. 25, 33615 Bielefeld, Germany*

Abstract

In this article, we explore the potential contribution of multimodal context information to object detection in an "intelligent car". The used car platform incorporates subsystems for the detection of objects from local visual patterns, as well as for the estimation of global scene properties (sometimes denoted "scene context" or just "context") such as the shape of the road area or the 3D position of the ground plane. Annotated data recorded on this platform is publicly available as the "HRI RoadTraffic" vehicle video dataset, which forms the basis for this investigation.

In order to quantify the contribution of context information, we investigate whether it can be used to infer object identity with little or no reference to local patterns of visual appearance. Using a challenging vehicle detection task based on the "HRI RoadTraffic" dataset, we train selected algorithms ("context models") to estimate object identity from context information *alone*. In the course of our performance evaluations, we also analyze the effect of typical real-world conditions (noise, high input dimensionality, environmental variation) on context model performance.

As a principal result, we show that the learning of context models is feasible with all tested algorithms, and that object identity can be estimated from context information with similar accuracy as by relying on local pattern recognition methods. We also find that the use of *basis function representations* [1] (also known as "population codes") allows the simplest (and therefore most efficient) learning methods to perform best in the benchmark, suggesting that the use of context is feasible even in systems operating under strong performance constraints.

Keywords: Real world systems, learning, object detection

1. Introduction

As the performance of computer hardware grows, it is becoming possible to construct autonomous systems of ever increasing complexity. Prominent examples for this development are, e.g., humanoid robots [2, 3] and intelligent vehicles [4, 5]. Numerous ways exist for coping with this additional complexity, such as formal hierarchical design languages [6], component-based graphical development systems [7] or machine learning methods [8, 9]. Learning methods are theoretically well-established and have resulted in huge advantages both w.r.t. performance as well as required design effort. A good example for this is the ubiquitous use of trainable object classifiers in visual object detection [4, 10, 11, 12, 13, 14, 15]: such methods transform low-level sensory information into symbolic quantities such as distinct object categories. Although the precise form of the learned decision heuristics is thus outside direct human control, results are of high quality and the design effort is strongly reduced.

In this contribution, a similar goal is pursued although on a different level of abstraction. In contrast to many "low-level" learning approaches, focusing on learning close to sensory signals [10, 11, 12, 13, 14, 15], this contribution explores the possibilities of learning on a higher abstraction level

*Corresponding author:
alexander.gepperth@ensta-paristech.fr

which we term *system-level*. This approach is motivated by models of information processing in humans and primates which emphasize the role of processing at very high abstraction levels, be it in the hippocampus or in "semantic hubs" close to the hippocampus [16]. A popular model even counter-intuitively suggests that the learning of new concepts may start at high abstraction levels, and that newly formed concepts are propagated in a top-down fashion in the processing hierarchy [17]. From a practical point of view, it stands to reason that learning at the system level has favorable properties: first, the data on which learning algorithm operate are of lower dimensionality, and secondly, the data are less subject to noise due to preceding processing stages.

1.1. The system-level learning paradigm

We use the term "system-level learning" (SLL) to describe a learning paradigm for agents operating under real-world conditions. This paradigm is defined by the high invariance and abstraction level of the processed data. Prerequisites are the existence of sensory (pre-)processing subsystems which transform high-dimensional, noisy sensor data into largely invariant and low-dimensional quantities, which we term *system-level quantities* (SLQ). Since sensory processing subsystems analyze different aspects of the same external world, we expect significant dependencies between their end-products, the SLQs. The basic claim of system-level learning is therefore that strong gains can be obtained by exploiting the dependency structure of SLQs, and that the analysis of such dependency structures is a worthwhile endeavour *in addition* to evaluating the SLQs themselves.

System-level learning is then concerned with the creation of dependency models (which we term "system-level models" or "context models") between SLQs. This is basically a supervised learning process which involves the estimation of certain SLQs from others. Since system-level learning is embedded into autonomous agents, we assume that a sufficient number of training examples can always be generated. However, this embedding also produces several constraints which will be discussed in the following section.

1.2. Elaboration of constraints imposed by system-level learning

In order to be applicable for system-level learning, any learning method must fulfill several requirements: online regression, generality and simplicity, scalability and data mining ability.

Online regression, i.e, the ability to train and perform regression (in contrast to binary classification) where the number of training examples is not known in advance. We demand regression ability because SLQs may not be binary quantities; online capability is required to adapt system-level models to changing statistics, and because it is impractical to specify the number of training examples in advance.

Generality and simplicity: Any system-level learning algorithm should be generic, i.e., not making too restrictive assumptions about the data it operates on. Similarly, it needs to be *simple* in the sense that it does not contain a large number of parameters that must be tuned to problem-specific values.

Scalability. In order to create system-level models, SLQs must be converted to a common representational format (see, e.g., [18]). Since SLQs may be very diverse, such a format cannot be optimized for a specific type of data. Therefore, the number of data dimensions can grow very large, a fact that suitable learning algorithms must be able to cope with.

Data mining ability. A learning algorithm must be able to ignore irrelevant and possibly noisy dimensions and to approximately identify the relevant ones, especially in high-dimensional data.

1.3. Messages of this article

First of all, the presented experiments are meant to demonstrate the value of context information in real-world object detection, showing a promising way to avoid the ambiguities of local pattern recognition methods in complex environments.

With the concept of system-level learning, we present a comprehensive approach for the acquisition of context models in autonomous agents, promoting the view that such models can be acquired automatically from data samples. By analyzing and comparing the performance of different learning algorithms under a variety of typical real-world influences, we ensure that our findings are not random artifacts of a particular algorithm or dataset.

Furthermore, we argue that the use of high-dimensional *basis function representations* (or "population encoding") supports the use of very simple, quasi-linear methods for learning context

models, which aligns well with the constrained nature of processing in autonomous agents[1].

Lastly, the experiments presented here can of course be considered a field study about the feasibility of certain learning algorithms in a challenging learning task under typical real-world conditions.

1.4. Background and related work

In this section, we shall review several learning algorithms and methods that may be suitable for system-level learning, as well as a number of articles that make use of context information for object detection.

Learning algorithms. The multilayer perceptron (MLP) model trained by an offline backpropagation algorithm (see, e.g., [19, 20]) is an established general-purpose learning algorithm. Although it can be used in an online fashion, MLP faces known issues of catastrophic forgetting in this case [19, 20], which makes its use for system-level learning problematic.

Several models have been proposed to remedy this issue. For example,in [21] the authors propose a mechanism based on biological studies about the hippocampus and the neocortex, using complementary learning networks.

LWPR [22] is explicitly designed to be an online method for high-dimensional data, and it avoids catastrophic forgetting by incrementally partitioning the input space into volumes to which individual linear regression models are applied. Although the usage of LWPR is not trivial and contains several parameters of unclear impact, it is a potentially suitable candidate for system-level learning.

Another related algorithm is the Radial Basis Function Network (RBFN) model [20] which filters the input data through a set of Radial Basis Functions (RBFs) and subsequently performing linear regression. The RBF centers and widths are chosen by a problem-specific optimization. This generates a large number of parameters, which violates the requirements of generality and simplicity (Sec. 1.2). We therefore consider the RBFN model to be incompatible with our requirements.

Support vector machines (SVMs, see e.g., [8]) were not considered because they violate several requirements put forward in 1.2. Although their performance in terms of classification is generally impressive, they cannot (easily) perform online learning, and regression is problematic as well. Furthermore, for best performance, the choice of kernel parameters must be tailored to the specific problem. Especially SVMs with explicitly problem-specific kernels such as, for example, [23], are inapplicable here.

Logistic regression (LR) is a standard algorithm in machine learning (see, e.g., [9] for details). Although a very simple algorithm, it is a candidate method for system-level learning since it possesses online regression capabilities and can presumably cope with both high dimensionality and noise.

Basis function encoding. Basis function encoding [1, 24], usually referred to as population (en)coding, serves to realize a common representational format (see [18] for details). Being a biologically motivated way of representing data, it employs discrete neuron-like elements which can carry "activity" (modeling neural firing rates) within a certain range, i.e., $[0, 1]$. Each neuron is assigned its own distinctive basis vector (or basis function) which defines its preferred stimulus, i.e., the stimulus the neuron responds to most strongly. Neural ensemble activity is usually defined by projecting an input vector onto the set of all basis functions. Typically, the basis functions form a strongly overcomplete set and are assigned such that nearby neurons will have similar basis functions. It follows that an interpretation of a population code is intimately tied to a knowledge of the basis functions. However, while basis functions may be domain-specific, the resulting set of neural activities is not, as it has an identical form regardless of the nature of the encoded input vector. As elaborated in Sec. 1.2, this is a requirement if potentially diverse SLQs are to be operated on by a single learning algorithm. Population coding can also be viewed as a way of representing probabilistic information because the topologically arranged set of bounded activities strongly resembles a probability distribution. In fact, numerous approaches exist that show how Bayesian inference can be naturally implemented using such a representational format [25].

Context information in object detection. The coupling of object detection and contextual information mediated by low-level modulation is demonstrated in [27] where "gist", a low-dimensional vi-

[1] To clarify this: the involved system-level quantities are intrinsically low-dimensional. However, their projection to a high-dimensional space significantly facilitates learning.

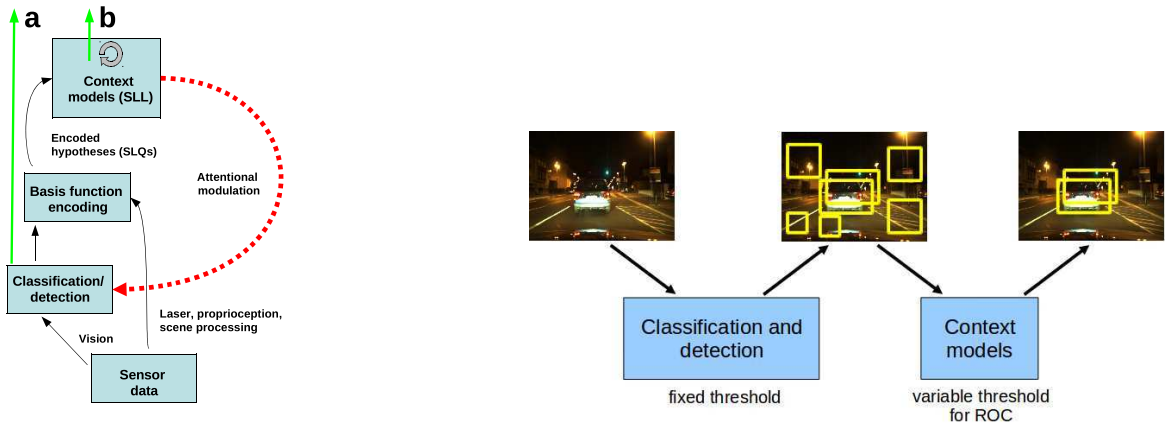

Figure 1: Two different ways of incorporating context models into SamSys [5, 26]. Left: data flow schematics for both cases, either attentional modulation of low-level processing (case **a**) or purely high-level hypothesis selection (case **b**). Case **a** is described in [26], whereas case **b** is investigated in this publication. In case **a** (attentional modulation), context models act directly on the object detection module (dashed red arrow), which is why resulting object hypotheses are directly taken from there. In case **b**, context models filter incorrect hypotheses from the object detection module which is now operating without influence from context models. Right: operation of high-level hypothesis selection (**a**) in detail. With a low threshold, object detection produces many hypotheses, most if them incorrect. Ideally, such hypotheses are suppressed based on context models, without any reference to the low-level pattern of appearance.

sual scene description, is used to infer the locations of relevant objects by statistical models. In this work, learning is achieved by computing statistics about the location and size of objects depending on gist in an offline fashion. The concept of gist is taken further in [28] where a generic probabilistic model of 3D scene layout is proposed. This model can be queried for likely locations of, e.g., cars or pedestrians in order to inform an exhaustive local object detector. This work is interesting because the images used to reason about 3D scene layout are actually monocular. Furthermore, object detection may not only be guided by global scene properties but also by other objects in the scene: in [29], a discriminative model of local object-to-object interaction is proposed that formalizes cooperation and competition between local detections of multiple object classes and gives a probabilistic interpretation of this process. This is different but not in contradiction to our approach, which emphasizes object-to-scene interactions, illustrating that there are multiple classes of non-local "context" information that may improve object detection. Lastly, object detection may also be regarded as an active process in which the performed gaze actions (i.e., object detections) should maximize information acquisition. Based on the saliency map approach of [30], a Partially Observable Markov Decision Process (POMDP) formalism is used in [31, 32] to optimize gaze target selection based on the detections arising from previous gaze targets, visual saliency and global scene priors. Lastly, [26] presents a method to translate context models into *attentional modulation maps* that are directly integrated with pattern-based object detection in a way reminiscent of the saliency map approach. System-level learning is used to train the used context models in exactly the same way as it is done in this article.

2. Embedding of context models into a real-world system

The investigations described in this article are based on SamSys, a large-scale vehicle detection system in road traffic environments [5, 26] which integrates multimodal information (laser, video) as well as a wide variety of vision-based subsystems such as visual object detection, stereo processing, visual tracking and free-area detection. At the top of the SamSys processing hierarchy, there is a module implementing system-level learning as outlined in Sec. 1.1. For each object hypothesis generated by SamSys, a set of system-level quantities (SLQs) is formed from the results of subsystem processing, and simple relations are learned between SLQs and the known category of the current object hypothesis: see Fig. 1(left). This category is derived

Figure 2: Example images taken from the 5 different streams of the HRI RoadTraffic dataset. during the recording of the dataset, care was taken to include a wide range of illumination and environment conditions, such as low sun (stream II), rain (stream III), night (stream IV) and snow (stream V).

from annotations during the training phase but is unavailable during performance.

On the one hand, the learned context models can be used for generating *attentional modulation signals* to the visual object detection subsystem, allowing directed search for certain object categories which results in more accurate and efficient detection performance. In this contribution, o we explore what happens when On the other hand, context models can be directly used as classifiers that accept or reject incoming hypotheses, as can be seen in Fig. 1 (right). It is this latter case that we explore here. For such an approach to be feasible, the hypothesis generation stage is parameterized to minimize the false negative rate (i.e., the number of missed objects) at the cost of many false detections (see Sec. 3 for details). The task of the context models is then to remove false detections while leaving correct detections untouched.

3. Benchmark task

All data used in this study can be taken or derived from the HRI RoadTraffic dataset (see [26] and Fig. 2), which is a publicly available set of high-resolution traffic video streams from a wide range of environment and weather conditions[2]. In addition to stereo video images, the dataset contains object annotations (mainly for vehicles and signs), ego-vehicle state information, as well as pre-computed stereo and free-area information. This information was added in order to allow other researchers to quickly reproduce SamSys results (see Sec. 2).

[2]To obtain the HRI RoadTraffic dataset, please write an email request to hri-road-traffic@honda-ri.de or to alexander@gepperth.net

All benchmark datasets described here are obtained from stream I (in the terminology of [26]) of the HRI RoadTraffic dataset unless otherwise mentioned. Actual data is generated using SamSys (see Sec. 2), its visual detection subsystem providing a number of visual object hypotheses. Since our approach requires object annotations, we perform hypothesis generation only for images with existing object annotations (roughly every tenth image). The detection threshold of the visual detection subsystem is set to a sufficiently low value so as to produce at least 40 hypotheses[3]. Each hypothesis is subsequently assigned an object identity of "vehicle" if it (mutually) intersects more than 80% of an existing vehicle annotation ("non-vehicle" otherwise). A set of 17 system-level quantities (SLQs)[4] is computed using the remaining SamSys subsystems (see [26, 33]), transformed to population codes of 16×16 neurons and subsequently concatenated into a single data sample. The size of 16×16 was chosen ad hoc to maintain a sufficient resolution both for one- and two-dimensional SLQs (see below). The assigned object identity (see above text) is encoded in SLQ 4), as sketched in Fig. 5. The used system-level quantities are described in Fig. 4. They are mainly based on monocular hypothesis properties (area, aspect ratio, center point position, center point y-position), stereo-based properties (distance to ego-car, height over ground plane) or multimodal properties (distance to road area). The ratio of positive ("vehicle") to negative ("non-vehicle") examples is approximately 1:5. Please see Fig.4 for an overview of SLQs used in the benchmark task.

The basic goal in all conducted experiments is to predict the SLQ for object identity from the set of all other SLQs. We created several datasets to investigate this problem: The *standard dataset* consists of 32000 samples. Each sample of the *standard dataset* consists of 17 population-coded SLQs, each of which containing 16x16 elements. One training example therefore has a dimensionality of $d = 17 \times 16 \times 16 = 4352$.

Since LWPR cannot deal with the high dimensionality of the standard dataset, we replace each of the 17 population-coded SLQs by the 2D coor-

[3]Experiments with more restrictive detection thresholds always led to worse context model performance

[4]The number 17 was not chosen for any particular reason. It was just the sum of all useful SLQs we could easily implement, plus some dummy quantities we included to show that learning is not affected by uninformative SLQs

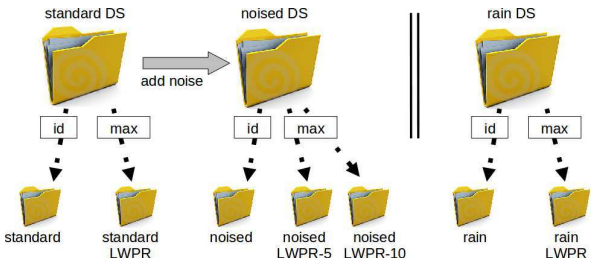

Figure 3: Used datasets

dinates of their respective maxima, giving 2 numbers for each SLQ. The dataset *standard LWPR* obtained in this way is an approximation of the *standard dataset* and has $17 \times 2 = 34$ dimensions. It is evident that this approximation incurs a loss of information and is therefore sub-optimal. We nevertheless employ it because is produces data of very low dimensionality which ensures good convergence of LWPR. Furthermore, the experiments of Sec. 5.2 indicate excellent performance of LWPR using the *standard LWPR dataset*, so the necessary information to solve the task is apparently preserved.

The *noised dataset* extends each sample of the *standard dataset* by 5 artificial "SLQs" where a Gaussian is generated at a random position in a 16x16 block. This simulates system-level noise, that is to say, uncorrelated results coming from other subsystems. We include these "unnecessary" dimensions to assess the data mining ability of the tested algorithms (see. 1.2). For use with LWPR, we create the dataset *LWPR-noised-10*, which has dimensionality 44, where 2x17=34 dimensions are taken from the *standard LWPR* dataset, and 10 dimensions contain uniformly distributed numbers between 0 and 15, representing maxima of 5 additional SLQs. For a study of LWPR under weaker noise, the dataset *LWPR-noise-5*, of dimensionality 39, is created in the same way but containing only 5 dimensions of noise.

The video stream I which was processed for creating the *standard dataset* was recorded during a sunny day. In order to check the generalization capability of all tested algorithms, we additionally create a dataset of 32000 samples based on a rainy-weather recording, and its LWPR approximation. These are called the *rain dataset* and *LWPR rain dataset*, and are generated in exact analogy to their standard dataset counterparts, except that the data source is stream IV of the HRI RoadTraffic dataset.

For an overview of used datasets and their dependencies, please see Fig. 3.

For the partitioning of samples into training and evaluation data, each dataset is subjected to a procedure called *blocking*: we group data into blocks of 1000 samples, each block corresponding to approximately 100 seconds of real time driving. Odd-numbered blocks are used for training, even-numbered blocks for testing. In real-world learning tasks, blocking allows to use qualitatively different data for training and testing and is a widely accepted procedure (see, e.g., [4]). Effectively, we therefore always use 16000 examples for training and 16000 examples for evaluation.

4. Methods

All algorithms discussed in this section are trained on the benchmark task described in Sec. 3. To speed up learning and testing, we apply a shortcut for the training of the multilayer perceptron, online multilayer perceptron and LWPR models: instead of predicting the population encoding of object identity (SLQ 4) in Fig. 4), these algorithms predict a single target value: 0.9 if object identity is "vehicle", 0.1 otherwise. We checked that this has no impact on results (as expected), while training times and memory consumption are strongly reduced. On the other hand, logistic regression was performed using the $16 \times 16 = 256$-dimensional population encoding of object identity as target quantity with negligible impact on training time.

4.1. Basis function encoding of system-level quantities

An SLQ is transformed into an activation pattern on a two-dimensional surface by using the techniques described in Sec. 1.4 (see also [1]), where the position and amplitude of the activation encode a (possibly two-dimensional) value distribution. Simple one or two-dimensional numeric quantities (e.g. distance, height or 2D position) are represented naturally by a localized peak; they are a special case of one-or two-dimensional distributions (e.g., histograms) which can be represented by a superposition of localized peaks (see also Fig. 4). This way of storing information is not optimized for storage efficiency, leading to high-dimensional data even when the intrinsic dimensionality of represented quantities is low. On the other hand, the representation of all SLQs in a common format allows the use of common learning algorithms regard-

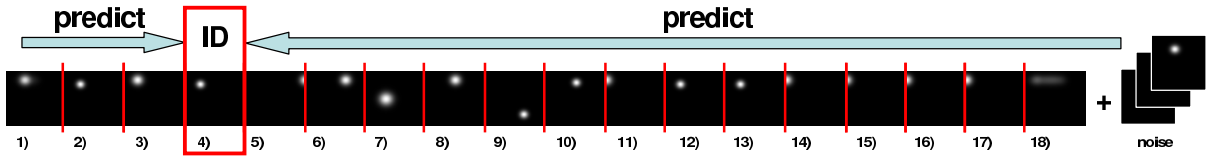

Figure 4: A typical data sample from the benchmark task, containing SLQs that are computed and population encoded using the methods of [26]. In general, one-dimensional quantities and histograms are encoded along the horizontal direction. The SLQs are: 1) 3D distance histogram 2) unused (constant), 3) 2D aspect ratio 4) assigned object identity based on ground-truth information, 5) 2D object area, 6) distance-to-free-area, computed from lower hypothesis border (see [26, 33]) 7) 2D position in image, 8) y-position in image, 9) physical size, 10-12) unused (constant) 13) distance-to-free-area, computed from center of hypothesis, 14) coordinate distance of closest point in 2D object hypothesis 15) difference of highest and lowest point in 2D object hypothesis 16-17) unused (constant) 18) height-over-ground-plane histogram. For SLQ 1), the histogram is taken over all pixels within the 2D hypothesis having a valid stereo distance. For SLQ 4), i.e., object identity, a Gaussian peak centered at $(4, 8)$ indicates a vehicle, whereas a peak at $(12, 8)$ indicates a non-vehicle. For SLQ 9), physical size is computed as the product of hypothesis width and hypothesis distance, which is itself taken from SLQ 14). For SLQ 18), the histogram is taken over all pixels within the 2D hypothesis having valid stereo information. Furthermore, in some experiments we add artificially generated "SLQs" which just consist of a randomly placed Gaussian peak. Given this set of population-coded SLQs, the goal of the benchmark task is to predict the value of SLQ 4), i.e., assigned object identity, from the remaining SLQs.

less of the origin of the data and is thus the basis for system-level learning.

4.2. Locally weighted projection regression

Locally Weighted Projection Regression (LWPR) is a method for learning high-dimensional function approximation based on the superposition of multiple linear models in the input space. The input dimensionality is reduced using a locally weighted variant of Partial Least Squares (PLS). We used the publicly available implementation of LWPR [22] by the authors for all described experiments. Since LWPR stores a covariance matrix for each used linear model, it cannot deal with very high-dimensional data of $d > 1000$ due to memory consumption.

LWPR is governed by several parameters and meta-parameters. First of all, it must be decided whether the covariance matrices should be only diagonal or whether non-diagonal matrices should be allowed. Furthermore, it must be decided whether an automatic step size adaptation for the linear models should be performed (this is termed meta-learning). Then, there are thresholds $\theta^{\mathrm{LWPR}}_{\mathrm{prune}}$ and $\theta^{\mathrm{LWPR}}_{\mathrm{create}}$ governing the creation and removal of receptive fields. Lastly, the initial receptive field size $\sigma^{\mathrm{LWPR}}_{\mathrm{init}}$ must be given.

In our experiments, default LWPR parameters were used except that meta-learning was enabled. Several initial receptive field sizes were tested, showing that smaller initial receptive field sizes result in the creation of a higher number of receptive fields, and also in higher precision. The main drawback of small receptive field sizes is that the dimensionality of the internal representations can quickly become too large for a computer simulation. LWPR results are obtained with an initial receptive field of $\sigma^{\mathrm{LWPR}}_{\mathrm{init}} = 1/32$, a value for which it performs well while avoiding the creation of too many

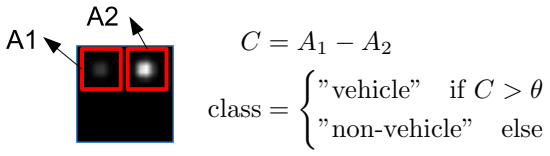

Figure 5: Decision making process for the decoding of estimates for SLQ 4) (see Fig. 4). We denote the sums over information-carrying areas by $A_{1,2}$. A vehicle is considered present (absent) in an example if $A_1 > (<) A_2$.

receptive fields. Step sizes for the training of linear models are chosen automatically, and the creation and pruning thresholds were set to $\theta^{\mathrm{LWPR}}_{\mathrm{prune}} = 0.8$, $\theta^{\mathrm{LWPR}}_{\mathrm{create}} = 0.3$. Standard practice for using LWPR is the offline mode which draws randomly chosen examples from the dataset until convergence of the prediction quality is observed. In contrast to this, we train LWPR in online mode by presenting training examples as they appear in the dataset. This is justified by the observation that the number, size and distribution of the receptive fields does not change much after the first iteration over the dataset in standard practice training. Instead of the standard, noised and rain datasets, we use their LWPR counterparts as described in Sec.3.

4.3. Logistic regression

Logistic regression (LR) is a quasi-linear standard algorithm in machine learning which makes certain assumptions about the data (see [9] for details). It is essentially a MLP with no hidden layer and a sigmoid activation function that is trained by gradient descent. LR is governed by a single parameter, the learning rate ϵ^{LR}.

We use the LR algorithm with 4352 input units, 256 output units and full connectivity from input

to output. The used learning rate is $\epsilon^{\mathrm{LR}} = 0.0001$. Population-coded output of dimensionality 256 is converted to graded confidence output by computing the quantity C as described in Fig. 5.

4.4. Multilayer perceptron

For providing a performance baseline by an algorithm whose suitability and performance has been shown in an overwhelming amount of studies, we train an offline MLP on the benchmark task described in Sec. 3. For this purpose, we ignore the fact that the MLP algorithm does not fully comply with the requirements outlined in Sec. 1.2. The MLP model [20] is a standard nonparametric regression method using gradient-based learning. The hidden layer(s) may be viewed as an abstract internal representation of the input, where it is however unclear what is being represented. For network training, we employ the back-propagation algorithm with weight-decay and a momentum term (see, e.g., [19]). This particular algorithm is parameterized by the number and size of hidden layers, the learning rate parameter ϵ^{MLP} and the momentum parameter ν^{MLP}. Our MLP has an input layer of size 4353 (1 bias element), one hidden layer of size 100 and one output neuron, applying a sigmoid non-linearity for hidden layer and output layer neurons. We verified that the results are similar for a number of hidden units between 50 and 200. Standard training of the MLP requires 5 *rounds* (gradient steps) before early-stopping[19] occurs (one round is one iteration over the whole dataset). Training convergence is fast in spite of the high input dimensionality. We work with a learning rate $\epsilon^{\mathrm{MLP}} = 0.01$ and a momentum parameter of $\nu^{\mathrm{MLP}} = 0.1$. We used the pyBrain-library [34] for all described MLP experiments.

4.5. The online multilayer perceptron model

We implemented an online MLP model which is trained incrementally: once it has been trained with a sample, we propagate that sample through the network again and evaluate the error between the output of the MLP and the expected output. If the error is higher than the memory threshold $\theta^{\text{O-MLP}}$, the sample is stored in a memory. When a new sample is received, the MLP is trained by this sample and also by all the samples hitherto stored in the memory. If the prediction error of a sample in the memory becomes lower than $\theta^{\text{O-MLP}}$, it is erased from the memory. This simple mechanism is intended to limit the effects of catastrophic forgetting. A compromise has to be found for the value of the threshold: if it is chosen too low, many samples will be stored, and the algorithm will be incapable of performing online. In contrast, if the threshold is chosen too high, the memory mechanism will not be able to affect the learning process. Summarizing, the online MLP algorithm is governed by the number of hidden layer units, the memory threshold $\theta^{\text{O-MLP}}$ and the learning rate $\epsilon^{\text{O-MLP}}$. No momentum parameter is used for this model. In our experiments, a memory threshold of $\theta^{\text{O-MLP}} = 0.2$ maintains a manageable memory size while keeping the system fast enough. We chose a learning rate of $\epsilon^{\text{O-MLP}} = 0.001$. Otherwise, all parameters are chosen identically to the multilayer perceptron model described above.

5. Experiments and Results

To establish a performance baseline for context models on the available benchmark datasets (standard, noised and rain datasets), we initially perform experiments with an MLP (see Sec. 4). The MLP model itself is not considered for system-level learning since it does not really comply with the constraints outlined in Sec. 1.2. In particular, this is the case because it is not an online method: it performs several gradient steps, each of which processes *all* training examples. Subsequently, we assess the performance of the remaining algorithms described in Sec. 4; these methods are all capable of online operation and fulfill the constrains of Sec. 1 to various degrees, which will be discussed in detail in Sec. 6.

Although the described experiments are conducted using different datasets, they always follow the blocking procedure (see Sec. 3) for splitting the used dataset into training and evaluation data.

5.1. Ground-truth data and evaluation measures

For evaluation, we employ a variable decision threshold θ^{dec} on the real-valued output of the various algorithms to obtain binary decisions. To determine the correctness of such decisions, the "true" object identity can be recovered from an evaluation example by decoding SLQ 4) as described in Fig. 5. By varying θ^{dec}, *receiver-operator characteristics* (ROCs) are obtained, giving the *false positive rate* fpr and the *false negative rate* fnr as implicit

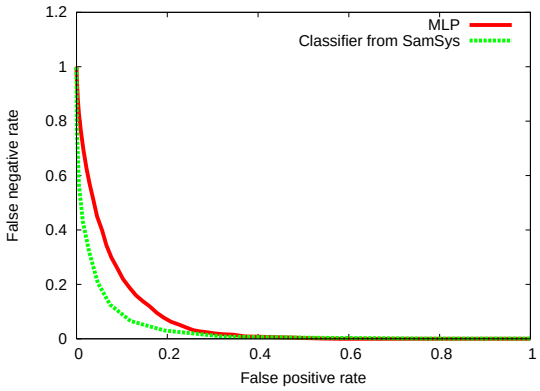

Figure 6: Vehicle classification performance of context models using an MLP. For comparison, we give the ROC obtained by the pattern-based object detection/classification subsystem of SamSys, rescaled such that a false positive rate of 1.0 corresponds to 40 incorrect hypotheses. Results are in a comparable range, which is remarkable considering context models disregard local pattern structure completely.

functions of θ^{dec}.

$$\text{fpr} = \frac{\#(\text{incorrect positive classifications})}{\#(\text{negative examples})} \quad (1)$$

$$\text{fnr} = \frac{\#(\text{incorrect negative classifications})}{\#(\text{positive examples})} \quad (2)$$

5.2. Baseline experiment

Fig. 6 displays, by means of a ROC as described in Sec. 5.1, the results of system-level learning with the MLP algorithm. Context model results using the standard dataset are compared to object detection results from the visual classification/detection hierarchy of SamSys (taken from [26]) operating on the same data, i.e., Stream I of the HRI RoadTraffic dataset. We can clearly observe that the performance of context models, although not better, is nevertheless in a comparable range when compared to local pattern-based detection.

5.3. Experiments with dedicated system-level learning algorithms

The *standard dataset* (see Sec. 3) is used to compare the basic performance of the investigated learning techniques: online MLP, LWPR and LR. Fig. 7 shows the result of this experiment. It clearly demonstrates that online MLP, LWPR and LR reach roughly equivalent results on the standard dataset. The best results are obtained by LWPR, which performs even better than the offline MLP model used as a reference (see previous section) and

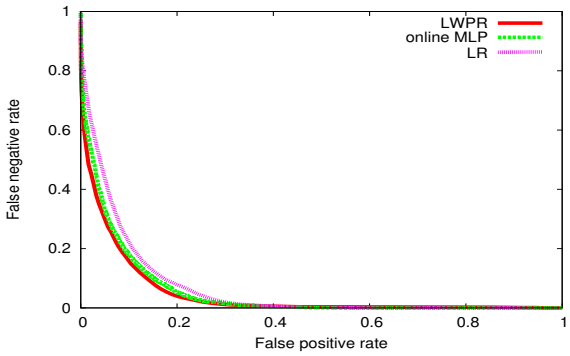

Figure 7: Performance of all tested algorithms on the standard dataset.

nearly reaches the performance of pattern-based object detection.

5.4. Resistance to noise

To investigate the resistance to system-level noise, i.e., irrelevant SLQs, we apply online MLP and LR to the *noised dataset*. Likewise, its two LWPR counterparts *noised LWPR-5* and *noised LWPR-10* are used to benchmark LWPR. Training and evaluation are conducted using the blocking procedure described in Sec 3. As can be observed on Fig. 8, the performance of online MLP decreases significantly while that of LR does not change at all. LWPR is obviously very sensitive to these added noise dimensions: the quality of the prediction decreases significantly when using the *noised LWPR-5* dataset and goes down to chance level for *noised LWPR-10*.

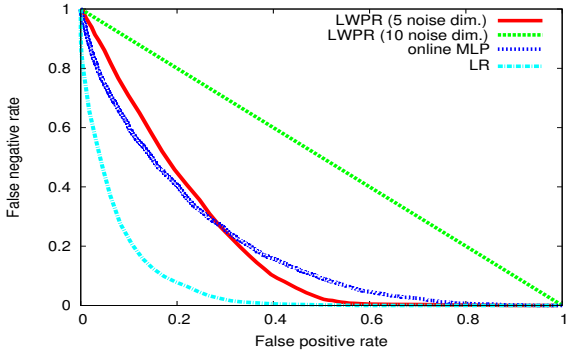

Figure 8: Performance of all tested algorithms under simulated system-level noise. Noteworthy are in particular the deterioration of LWPR and online MLP performances, and the robustness of LR.

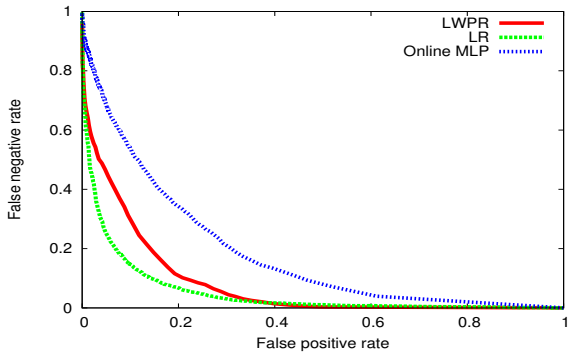

Figure 9: Performance of all algorithms on the *rain dataset*. Performance is slightly decreased compared to the *standard dataset*, which is most likely caused by more difficult environment conditions during the recording of the rain dataset. However, the results clearly show that learning is possible and feasible on the rain dataset, and that the fundamental conclusions of this study do not depend on the properties of the *standard dataset*.

5.5. Generalization capabilities

In order to assure that our results do not depend on a particular dataset, we evaluate all algorithms using the *rain dataset* and its LWPR counterpart. These datasets are analogous to the *standard (LWPR) dataset* but are obtained from data recordings on a rainy day (see Sec. 3). We first train and evaluate all algorithms on the *rain dataset* using again the blocking procedure described in Sec. 3 to obtain independent training and evaluation sets. Fig. 9 shows the results of this experiment. It can be clearly seen that the *rain dataset* is indeed more challenging than the *the standard dataset* since all algorithms, although they do give useful results, exhibit slightly impaired performance. This is consistent if one considers the fact that the performance of pattern-based object recognition also deteriorates under conditions of rain[26]. Now we wish to investigate whether context models can generalize across different environment conditions: for this purpose, we train all algorithms on the *standard dataset*, and evaluate them on the *rain dataset*. Fig. 10 shows the performance of online MLP, LR and LWPR on this task. As may be expected, we observe a further slight decrease in performance, which is again consistent with an equivalent experiment conducted for pattern-based object detection in [26]. The strong performance of online MLP in this experiment is surprising; further study will need to be applied to understand the reasons for this behavior. However, as online MLP is not a feasible candidate for system-level learning anyway due to its high sensitivity to noise (see previous section), we can safely disregard this result for the present.

5.6. Benefits of using population codes

As the LR algorithm has shown leading performance in the experiments presented up to now, we wish to investigate the reasons for its good performance. After all, the algorithm is almost linear and vastly less complex than, e.g., LWPR. Correspondingly, one would expect that it can learn less complex dependencies. To investigate this, the following experiments compare the performance of LR on the *standard dataset* and the *standard LWPR dataset*. We know that the latter dataset contains all information necessary for succcessful learning since LWPR performs well on it. As can be seen from Fig. 11, this is clearly not the case for LR. It can therefore be deduced that it is not so much the content but the proper *representation* of the data (here done by population encoding) that allows LR to reach its full performance.

6. Discussion

In this section we will summarize our results in order to subsequently discuss their value, plausibility and credibility, putting our methods and results into relation with related work in the literature.

6.1. Summary of results

We verified based on an instance of a large-scale real-world system that object detection can be performed without a detailed analysis of local patterns

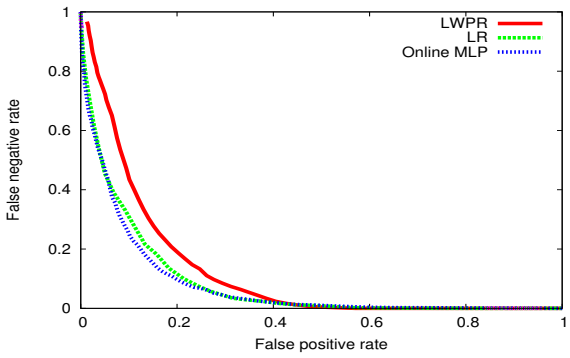

Figure 10: Test of generalization capability across environment conditions. Training was done on the *standard dataset*, evaluation on the *rain dataset*. All algorithms are able to generalize to different environment conditions, to various degrees.

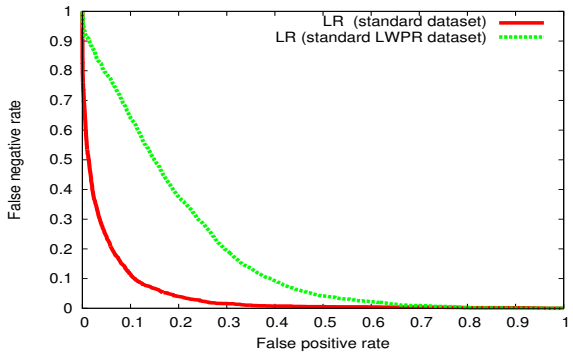

Figure 11: Benefits for simple learning algorithms by using population encoding of data. Plot shows performance of logistic regression using the high-dimensional *standard dataset* and the low-dimensional *standard LWPR dataset*. Performance on the *standard LWPR dataset* is strongly inferior even though it contains approximately the same amount of information (documented by the performance of the LWPR algorithm).

but solely based on what is commonly called "context information". We showed that results are of similar quality to those of a local pattern-based object detection method, and that context information can be accessed by the paradigm of system-level learning using very simple and efficient learning methods.

Generalizing this finding, we argue that dependencies between internal subsystems can be found and learned in *any* real-world processing system since events perceived by different subsystems often have a common or related cause. We therefore expect that the paradigm of system-level learning as presented here will be a way to achieve better performance, but also higher robustness, in the face of the complexity of real-world environments.

As a last point, we find that the transformation of system-level quantities (SLQs) (which form the basic building blocks of system-level learning) by a population encoding step is strongly beneficial for learning context models as demonstrated by the experiment of Sec. 5.6. Although the SLQs themselves are low-dimensional by construction, a projection to a higher-dimensional space facilitates the use of simple learning methods which are very efficient w.r.t. memory consumption and computation time. There may be two reasons for this: firstly, "blowing up" the dimensionality while leaving the intrinsic structure of the data unchanged allows to use more free parameters while the problem complexity remains unchanged. Secondly, many problems may be linearly separable in the high-dimensional space even if they are not separable in the original space. As the population encoding step is strongly non-linear, there is an interesting analogy to the way a support vector machine operates: nonlinear projection into a high-dimensional space followed by linear classification.

6.2. Critical discussion of system structure

In this study, we have used a rather naive way of coupling pattern-based object detection and context information, as the goal of the study was to isolate the contribution of context to object detection. As stated before, the primary goal of this study was to isolate and quantify the potential contribution of context information to object detection, not to present a finished object detection architecture. In contrast to [26], the presented coupling method is only able to reduce the number of false detections because the context models have no influence on hypothesis generation itself. For this reason, we chose a very low threshold for object detection since false detections can be eliminated by context models whereas missed detections are not. In all experiments, increasing this threshold invariably deteriorates context model performance as the reduction of false detections is accompanied by a strong increase of missed detections.

One may furthermore argue that the comparison between context models and pattern-based object detection presented in Sec. 6 is not completely fair since context models operate on hypotheses generated by pattern-based object detection (see Fig. 1). This implies that, on the one hand, context models may have an easier task to solve, but, on the other hand, that their performance depends on the received hypotheses: any missed detections will impair the performance of context models as well. For the first point, we point out that the provided hypotheses are almost arbitrary because a very low object detection threshold was chosen. From this, it follows that local pattern information enters only to a very small part into the processing chain. A more straightforward way to do this investigation would have been to present context models with randomly placed "hypotheses" while ensuring that all annotated objects from ground-truth data are included. Preliminary tests using this procedure were conducted with almost indistinguishable results. For the second point, we encountered very few cases where vehicles were missed, due the low detection threshold that was used. Taking everything into consideration, we believe that the com-

parison we did was fair to both compared methods. Furthermore, for the reasons given above, we believe our results do not depend to a significant extent on the employed object detection method.

6.3. Plausibility in the light of related work

First of all, we argue that the results presented in Sec. 5 are credible because they align well with what has been found for local pattern-based object recognition on the same data (see [26]). Not only are the results in a comparable range, but also the development of generalisation performance under changed environment conditions is very similar. For making the results more reproducible, we have made the HRI RoadTraffic dataset publicly available (see Sec. 3). Although we could not make available the pattern-based object detection system used to obtain hypotheses, its contribution to the experiments is in any case very small: as we verified, randomly selected hypotheses could be used just as easily. The fact that various types of context information are useful for object detection has been demonstrated in several publications [27, 28, 29, 32], even in real-world scenarios [28], so our results are not entirely unexpected.

Given the presented results, one might be inclined to question whether simple learning methods such as logistic regression may really capture the complexities of difficult object detection tasks. Although our results clearly confirm this, we can only speculate about the reasons for this success. In our view, the simple methods presented here work well due to the nature of the data that they operate on, which consist of system-level quantities (SLQs) that have already been processed extensively. As a consequence, SLQs are intrinsically low-dimensional even though population encoding greatly increases their dimensionality. Furthermore, it is known from related work that scene context often imposes very simple constraints on objects [4, 28, 35, 36]; it therefore stands to reason that simple algorithms can capture such constraints. We therefore argue that simple algorithms will produce feasible context models whenever they operate on low-dimensional, invariant quantities, which coincides with the fundamental assumptions of system-level learning as stated in Sec. 1.1.

6.4. Discussion of LWPR performance

A somewhat surprising fact is the unfavorable performance of LWPR under noisy conditions, see Sec. 5.4. From the basic mechanisms of receptive field generation within LWPR, is can be understood why random noise might be unfavorable. Nevertheless, there exist mechanisms to gradually align existing receptive fields to data statistics, which should result in better resistance to noise. However, this mechanism is quite slow and involves several parameters. It is possible that we did not set these parameters correctly, or that the mechanism simply did not have time to converge. In the latter case, several more iterations over the training data could have remedied the problem. However, such a type of learning could never be done in an embodied agent because the number of examples would have to be known in advance. More investigations about the behavior of LWPR under noise will be required to resolve this issue.

7. Conclusion and future work

In this article, we tried to make a strong point for the value of context information to object detection by showing its value even in the face of complex environments and adverse processing conditions. We believe, however, that the presented approach can be extended in many ways to make the best possible use of all information sources available to an intelligent vehicle. In particular, we plan to investigate the following topics: first of all, it will make sense to include additional sensors (radar, laser, PMD) into the system-level learning concept as additional sources of hypotheses. Furthermore, it is not really practical to use only predefined SLQs: an additional learning layer will be added that can derive meaningful SLQs from existing ones. A good example for this is the combination of measured object distance and retinal size into physical object size, which is a highly disciminative indicator for object identity, whereas the original quantities are not. Lastly, we would like to complement the "late rejection" appproach adopted here by a mechanism that can actively encourage the creation of hypotheses in specific locations, according to learned system-level models, as investigated in our previous work[26]. In this way, we hope to contribute to the creation of real-world perception system than can honestly be claimed to match human performance.

References

[1] A Pouget, P Dayan, and RS Zemel. Inference and computation with population codes. *Annu Rev Neurosci*, 26:381–410, 2003.

[2] B Bolder, H Brandl, M Heracles, H Janssen, I Mikhailova, J Schmuedderich, and C Goerick. Expectation-driven Autonomous Learning and Interaction System. In *IEEE-RAS Int. Conf. Humanoids*, 2008.

[3] J Schmuedderich, H Brandl, B Bolder, M Heracles, H Janssen, I Mikhailova, and C Goerick. Organizing Multimodal Perception for Autonomous Learning and Interactive Systems. In *IEEE-RAS International Conference on Humanoid Robots*, pages 312–319, Daejeon, Korea, December 2008.

[4] B Leibe, N Cornelis, K Cornelis, and L Van Gool. Dynamic 3d scene analysis from a moving vehicle. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007.

[5] J Schmüdderich, N Einecke, S Hasler, A Gepperth, B Bolder, S Rebhan, M Franzius, R Kastner, B Dittes, and H Wersing. System approach for multi-purpose representations of traffic scene elements. In *Proceedings of the ITSC*, 2010.

[6] C Goerick. Towards an understanding of hierarchical architectures. *Transactions on Autonomous Mental Development*, 3, 2010.

[7] A Ceravola, M Stein, and C Goerick. Researching and developing a real-time infrastructure for intelligent systems. *Robotics and Autonomous Systems*, 2007.

[8] V Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2 edition, 2000.

[9] CM Bishop. *Pattern recognition and machine learning*. Springer-Verlag, New York, 2006.

[10] H Wersing and E Körner. Learning optimized features for hierarchical models of invariant object recognition. *Neural Computation*, 15(7), 2003.

[11] D Walther, L Itti, M Riesenhuber, T Poggio, and C Koch. Attentional selection for object recongition - a gentle way. In *Lecture Notes in Computer Science*, volume 2525. Springer, 2002.

[12] A Gepperth, J Fritsch, and C Goerick. Computationally efficient neural field dynamics. In *Proceedings of the 16th European Symposium on Artificial Neural Networks*, pages 179–185, 2008.

[13] S Wiegand, C Igel, and U Handmann. Evolutionary optimization of neural networks for face detection. In *12th European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 139–144. Evere, Belgium: d-side publications, 2004.

[14] Y LeCun, FJ Huang, and L Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[15] P Viola and M Jones. Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 1:511–518, 2001.

[16] K Patterson, P Nestor, and T Rogers. Where do you know what you know? the representation of semantic knowledge in the human brain. *Nature Reviews Neuroscience*, 8(12):976–987.

[17] S Hochstein and M Ahissar. View from the top: hierarchies and reverse hierarchies in the visual system. *Neuron*, 36(5):791–804, Dec 2002.

[18] A Gepperth, J Fritsch, and C Goerick. Cross-module learning as a first step towards a cognitive system concept. In *Proceedings of the First International Conference On Cognitive Systems*, 2008.

[19] RJ Reed and RJ Marks II. *Neural smithing: Supervised Learning in Feedforward Artificial Neural Networks*. MIT Press, 1999.

[20] S Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall, 1999.

[21] M Hattori. Avoiding catastrophic forgetting by a dual-network memory model using a chaotic neural network. *World Academy of Science Engineering and Technology*, 12 2009.

[22] S Vijayakumar and S Schaal. Locally weighted projection regression: Incremental real time learning in high dimensional space. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1079–1086, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[23] B Mersch, T Glasmachers, P Meinicke, and C Igel. Evolutionary optimization of sequence kernels for detection of bacterial gene starts. *Int J Neural Syst*, 17(5), 2007.

[24] DC Knill and A Pouget. The bayesian brain: the role of uncertainty in neural coding and computation. *Trends Neurosci*, 27(12), 2004.

[25] WJ Ma, JM Beck, PE Latham, and A Pouget. Bayesian inference with probabilistic population codes. *Nature Neuroscience*, 9(11), 2006.

[26] ART Gepperth, S Rebhan, S Hasler, and J Fritsch. Biased competition in visual processing hierarchies: A learning approach using multiple cues. *Cognitive Computation*, 3(1):146–166, 2011.

[27] K Murphy, A Torralba, D Eaton, and WT Freeman. Object detection and localization using global and local features. In J Ponce, editor, *Toward Category-Level Object Recognition*, Lecture Notes in Computer Science. Springer, 2005.

[28] D Hoiem, AA Efros, and M Hebert. Putting objects into perspective. *International Journal of Computer Vision*, 80(1), 2008.

[29] C Desai, D Ramanan, and C Fowlkes. Discriminative models for multi-class object layout. In *International Conference on Computer Vision (ICCV)*, 2009.

[30] L. Itti and C. Koch. Computational modelling of visual attention. *Nat Rev Neurosci*, 2(3):194–203, Mar 2001.

[31] J Vogel and O De Freitas. Target-directed attention: Sequential decision-making for gaze planning. In *International Conference on Robotics and Automation (ICRA)*, 2007.

[32] J Vogel and K Murphy. A non-myopic approach to visual search. In *Computer and Robot Vision*, volume 0, pages 227–234, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[33] A Gepperth. Implementation and evaluation details of a large-scale object detection system. Technical Report TR 10-11, Honda Research Institute Europe GmbH, 2010.

[34] T Schaul, J Bayer, D Wierstra, S Yi, M Felder, F Sehnke, T Rückstieß, and J Schmidhuber. Pybrain. *Journal of Machine Learning Research*, 2010.

[35] M. Szczot, O. Lohlein, M. Serfling, and G. Palm. Incorporating contextual information in pedestrian recognition. In *Proc. of the IEEE Symposium on Intelligent Vehicles*, 2009.

[36] M Enzweiler and D Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12), 2009.